Position Paper: Virtual Communications Backplane

Terry Jones[1], Scott Atchley[1], Geoffroy Vallee[1], Joshua Hursey[2], Yoav Tock[3], Benjamin Mandler[3], Eliezer Dekel[3]

{trj, atchleyes, valleegr} @ ornl.gov[1]
jhursey @ uwlax.edu[2]
{tock, mandler, dekel} @ il.ibm.com[3]

Problem statement:
Today, the HPC execution environment is both heavily layered for software engineering reasons and heavily dependent upon decisions based-on data held in multiple layers. This is especially true of extreme-scale parallel workloads: the operating system kernel layer is largely unaware of important remote status; the distributed parallel runtime system layer is largely unaware of relevant details held by kernels and file systems, and the compiler systems lack information helpful for their role. Moreover, the limited remote data that is available is often too out-of-date for many useful decisions. Finally, emerging energy concerns dictate that the cost of data movement must be carefully weighed against the benefit, and that work must proceed with minimal movement to meet power constraints. The resulting communications infrastructure on HPC systems is a hodge-podge of fractured independent communication stacks. These facts are at odds with the emergence of HPC architectures consisting of extreme levels of concurrency and complexity.

Proposed approach:
We propose an innovative communications framework to facilitate extreme-scale environments and obviate the need for separate vertically-integrated stacks. Today, separate ad hoc stacks are maintained for many critical components including parallel file system communication, job-launch communication, system monitoring communication, application tool communication, various heart-beat and flow control communications, and so on. Indeed, the current 'duplication of effort' approach places a huge burden on their (possibly small) development teams to ensure the necessary best-of-breed characteristics of extreme scalability, resilience, flexibility and so forth are well suited to rapidly evolving exascale environments. We propose a virtual communications backplane designed to provide the required fundamental infrastructure and associated tools. To eliminate the present (yet undesirable) vertically-integrated approach for each independent need, our proposed solution provides a communications capability featuring a peer-to-peer and overlay capability built over an efficient network abstraction layer, and incorporating both fault resilience and parallel aware interfacing (for adaptive operating systems and runtime systems) and based on a minimal network abstraction layer optimized for performance.

The Virtual Communications Plane is a bottom-up approach designed for wide adoption, building on the success of CCI. Adding basic functionality that can support fault tolerance (e.g., reliable multicast) will help higher level services build more complex services without duplicating too much effort. Key to our design will be an avoidance of pushing too much into the network abstraction layer, and cleanly support the needs requested by the community at the higher level of the virtual control plane interface as a second stage.

The communication plane concept supports the notion of independent community requirements through separate planes. For example, it is possible to provide a *control plane* separate from a *data plane*; yet each maintains the same network abstraction layer. We see a similar trend in two other areas of IT: the cloud, where a huge data center is controlled to provide diverse service to clients (computation, storage, application hosting, etc); and software defined networks (SDN) where a central or distributed control system is controlling and managing the setup, configuration,

and services provided by an enterprise network. In both cases the division to control plane and data plane is manifest. The resulting communications infrastructure will support the characteristics needed for an integrated environment. Chief among these requirements in realizing high payoff-potential are: timely – the information is *recent* enough to meet the need; relevant – the *desired* info is made available; resilient – the design must be able to *continue* in the presence of faults; and adaptive – the design must quickly react to system/workload changes like load imbalances and failed resources.

Related Work:
CCI [1,2,3], SpiderCast [4,5,6,7,8], CIFTS improved fault tolerance [9,10], fault-tolerance schemes [11,12,13]. In our prior CCI work, we have support for CCI over sockets (our reference implementation), OFA Verbs (InfiniBand and RoCE), Cray GNI (Gemini, Aries), and Cray Portals (SeaStar). Collaborators are working on a Linux Native Ethernet (IP-bypass) (Brice Goglin, the developer of Open-MX) and a fully OS-bypass, zero-copy implementation (Myricom and Emulex). Projects like CIFTS have approached this problem from the top-down by working with various high-level service that share 'events' over an unreliable communication service call the Fault Tolerant Backplane (FTB) [Disclaimer: our team includes CIFTS members]. Although some services demonstrated prototypes using the FTB, it struggled in wider adoption due to the complexity of the interface and the unreliable nature of the communication service. We propose a bottom-up approach using a communication service with adjustable reliability that builds on the success of the CCI project and new capabilities from the SpiderCast project.

Assessment:
The provision of virtual communications backplane will enable novel programming and infrastructure approaches and will impact many important topics necessary for extreme scale computing. For example, tools will greatly benefit from the available information and infrastructure. Operating systems and runtime systems can make decisions with parallel awareness and expose data with much greater efficiency. Compiler systems will have two-way communication access with data required to support advances in high-concurrency and adaptive strategies. Finally, emerging needs such as energy-efficiency and resilience decisions can be made on a global view instead of with limited scope.

- Challenges addressed: resilience, parallelism and OS/runtime structure (reduced competing comm), applications structure (improved comm).
- Maturity: Our approach offers much in terms of maturity. As for maturity of CCI, the API is stabilizing now that we have implementations on multiple networks. Work is on-going for performance tuning and error-handling as well as adding new capabilities on existing networks (i.e. zero-copy and OS-bypass over Ethernet). We have discussed CCI over PAMI with IBM and it would not be a difficult port (one could think of CCI as a subset of PAMI), but neither ORNL or IBM has resources to commit to a port at this time.
- Uniqueness/Novelty: Would be unique (needs currently accomplished as individual vertically integrated stacks).
- Applicability: Possibly applicable to large parallel data-mining and/or database environments.
- Effort: Proposed small to medium size team for 3 years.

References:
[1] Scott Atchley, David Dillow, Galen Shipman, Patrick Geoffray, Jeffrey M. Squyres, George Bosilca and Ronald Minnich, “The Common Communication Interface (CCI)” in

the *19th IEEE Symposium on High Performance Interconnects (HOTI),* Santa Clara, CA, August 23-25, 2011.

[2] http://cci-forum.com

[3] CCI API manual, http://cci-forum.com/wp-content/uploads/2012/06/cci-manual-0.1b1.pdf

[4] G. Chockler, R. Melamed, Y. Tock, and R. Vitenberg, "Spidercast: a scalable interest-aware overlay for topic-based pub/sub communication," in *DEBS '07: Proceedings of the 2007 inaugural international conference on Distributed event-based systems*. New York, NY, USA: ACM, 2007, pp. 14-25.

[5] Gregory Chockler, Sarunas Girdzijauskas, Roie Melamed, Yoav Tock, Ymir Vigfusson. Magnet: Practical Subscription Clustering for Internet-Scale Publish/Subscribe. In *DEBS 2010, 4th ACM International Conference on Distributed Event Based Systems*.

[6] Yoav Tock and Benjamin Mandler. SpiderCast: Distributed Membership and Messaging for HPC Platforms: An Architectural Overview and High Level Design. IBM Technical Report IBM-IL-YT2010-1. Jan-2010. Haifa, Israel.

[7] Yoav Tock, Benjamin Mandler, and Gennady Laventman. SpiderCast: Distributed Membership and Messaging for HPC Platforms: Publish-Subscribe and DHT Services High Level Design. IBM Technical Report IBM-IL-YT2010-2. May-2010. Haifa, Israel.

[8] Y. Tock, B. Mandler, J. Moreira, and T. Jones. Poster: Scalable Infrastructure to Support Supercomputer Resiliency-Aware Applications and Load Balancing. *In companion to International Conference for High Performance Computing, Networking, Storage and Analysis (Poster, SC'11)*. Nov-2011. Seattle, WA.

[9] Gupta, R.; Beckman, P.; Park, B.-H.; Lusk, E.; Hargrove, P.; Geist, A.; Panda, D.K.; Lumsdaine, A.; Dongarra, J.; , "CIFTS: A Coordinated Infrastructure for Fault-Tolerant Systems,". *ICPP '09. International Conference on Parallel Processing, 2009*, vol., no., pp.237-245, 22-25 Sept.2009 doi:10.1109/ICPP.2009.20
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5362328&isnumber=5361798

[10] CIFTS Project webpage: http://www.mcs.anl.gov/research/cifts/

[11] Hursey, J.; "Coordinated Checkpoint/Restart Process Fault Tolerance for MPI Applications on HPC Systems," Ph.D. Dissertation, Indiana University. July 2010
http://proquest.umi.com/pqdweb?did=2173787631&sid=1&Fmt=2&clientId=12010&RQT=309&VName=PQD

[12] Hursey, J.; Squyres, J.M.; Mattox, T.I.; Lumsdaine, A.; , "The Design and Implementation of Checkpoint/Restart Process Fault Tolerance for Open MPI," *IEEE International Parallel and Distributed Processing Symposium (IPDPS) Workshop on Dependable Parallel, Distributed and Network-Centric Systems (DPDNS)*. pp.1-8 March 2007
http://dx.doi.org/10.1109/IPDPS.2007.370605

[13] Hursey, J.; Graham, R.; "Analyzing Fault Aware Collective Performance in a Process Fault Tolerant MPI," Journal of Parallel Computing. 38, 1-2, pp.15-25 January 2012

URL: http://www.sciencedirect.com/science/article/pii/S0167819111001414